APPARATUS AND METHOD FOR USER-DEFINED TUNABLES

Technical Field

The technical field is UNIX® operating systems. More specifically, the technical field relates to tools and processes that a system administrator can employ to create tunables, and to adjust the values assigned to tunables.

Background

One central component of a computer system operating in a UNIX® environment is an operating system kernel. In a typical UNIX® environment, many applications, or processes, may be running. All these processes use the memory-resident kernel to provide system services. The kernel manages the set of processes that are running on the system by ensuring that each such process is provided with some central processor unit (CPU) cycles when needed, and by arranging for such process to be resident in memory so that the process can run when required. The kernel provides a standard set of services that allows the process to interact with the kernel. In the UNIX® environment, these services are sometimes referred to as system calls because the process calls a routine in the kernel to undertake some specific task. The kernel will then perform the task, and will return a result. In essence, the kernel fills in the gaps between what the process intends to happen and how system hardware needs to be controlled to achieve the process's objective.

The kernel's standard set of services is expressed in a set of kernel modules (or simply, modules). The kernel typically includes modules such as drivers, file system modules, scheduling classes, Streams modules, and system calls. These modules are compiled and subsequently linked together to form the kernel. When the system is started, or "booted up," the kernel is loaded into memory.

The UNIX® environment often employs kernel modules having adjustable parameters. Such adjustable parameters are commonly known as tunables. In current systems, tunables are created, and their default values are set by the kernel developer. In these current systems, the computer system end-user, or system administrator, is able to adjust the values assigned to these tunables, but is not able to create new tunables.

Summary

What is disclosed is a user-defined tunable that includes a tunable name, a tunable description, an assigned value, and expressions that relate one or more kernel tunables to the user-defined tunable. Each of the kernel tunables includes a parameter value defined

by an expression. A change to the value of the user-defined tunable changes the parameter value of each of the kernel tunables whose values are expressions involving that user-defined tunable.

Also disclosed is an apparatus that provides user-defined tunables for use in a UNIX® operating system. The apparatus includes a system administrator interface having a user-defined tunable creation option, and a system administrator controlled value assignment option. The apparatus further includes a tunable repository that stores the user-defined tunables, and kernel configuration tools that read the user-defined tunables from the tunable repository and relate the user-defined tunables to a kernel in the UNIX® operating system.

Further, what is disclosed is a method for implementing user-defined tunables in a UNIX® operating system. The method includes the steps of creating a user-defined tunable and, using an expression, relating the user-defined tunable to one or more kernel tunables.

Finally, what is disclosed is a computer-readable medium having code to implement user-defined tunables in a UNIX® operating system. When implemented, the code allows performance of the steps of creating a user-defined tunable and, using an expression, relating the user-defined tunable to one or more kernel tunables.

Description of the Drawings

The detailed description will refer to the following figures in which like numerals refer to like items, and in which:

Figure 1 illustrates a kernel having a number of kernel modules;

Figure 2 is a block diagram of an embodiment of a UNIX® environment in which user-defined tunables are implemented;

Figure 3 shows an embodiment of the system administrator interface used in the UNIX® environment of Figure 2;

Figure 4 is a logical diagram of a user-defined tunable;

Figure 5 is a flowchart illustrating a process for implementing user-defined tunables; and

Figure 6 shows a computer system using a UNIX® operating system, and having the feature of user-defined tunables.

Detailed Description

In a UNIX® operating environment, kernel modules are provided with developer-provided tunables (referred to hereafter as kernel tunables). The operating system kernel

may have a number of kernel modules. Each kernel module may define any number of kernel tunables, with the kernel tunables having tunable parameters, which are integer variables that control the behavior of the module. Kernel tunables are used for a variety of different tasks: some kernel tunables control resource allocations, other kernel tunables control security policies, still other kernel tunables enable optional kernel behavior. A typical kernel may have as many as 200 kernel tunables.

Figure 1 illustrates a kernel 10 having a number of kernel modules 11. Each kernel module 11 includes kernel code 12 and a modmeta file 13. The modmeta file 13 includes tunable parameters 14. The tunable parameters 14 define the kernel tunable. The tunable parameters 14 are initialized by a tunable initialization function, which is part of the kernel code 12. A tunable handler function 15 controls the tunable parameters 14. Some tunable parameters 14 are set by the kernel developer, and cannot easily be changed subsequent to installation of the operating system kernel. Furthermore, the system administrator cannot create kernel tunables.

The kernel 10 also incorporates user-defined tunables (not shown). User-defined tunables are those created by system administrators. One user-defined tunable can be related to another user-defined tunable by way of expressions. User-defined tunables are also used in expressions for one or more kernel tunables and, hence, act as scaling factors. Once these user-defined tunables are established by the system administrator, the system administrator can change one user-defined tunable to modify all related kernel tunables. For example, the system administrator could create a num_databases tunable, and then set several kernel tunables based on its value. A subsequent change to the value of num_databases causes all related kernel tunable values to change as well. In addition, user-defined tunables can be defined by system administrators using tunable names that make sense to the system administrator. Finally, the user-defined tunables act exactly like kernel tunables in kernel configuration tools. Thus, user-defined tunables make the job of kernel tuning easier for the system administrator.

Figure 2 is a block diagram of an embodiment of a UNIX® environment 100 in which user-defined tunables are implemented. The environment 100 includes core kernel 110. Associated with the core kernel 110 is tunable repository 120, in which user-defined and kernel tunables are provided. Kernel module developer interface 140 is used to create the kernel tunables that are provided in the tunable repository 120. User/administrator interface 130 is used to create the user-defined tunables that are provided in the tunable repository 120. System administrator interface 150 is used to add

expressions to relate user-defined tunables and kernel tunables. The system administrator interface 150 may also be used to change user-defined tunables and to assign values to kernel tunables. In an embodiment, the user/administrator interface 130 and the system administrator interface 150 are the same interface. Finally, kernel configuration tools 160 are used to relate the kernel and user-defined tunables to the core kernel 110. More specifically, the kernel configuration tools read the definitions of the tunables provided in the tunable repository 120, evaluate the expressions that relate the tunables, and provide results to the core kernel 110.

The system administrator interface 150 includes the necessary tools to view the user-defined tunables, to change tunable parameters, and to create expressions that relate the user-defined tunables to kernel tunables. Figure 3 shows an embodiment of the system administrator interface 150. The system administrator interface 150 includes a number of tools that can be used by a system administrator to query and change the values of the user-defined tunables and to change the expressions that relate the user-defined tunables to other tunables. The tools include commands and options that the system administrator uses to create and change user-defined tunables and to change kernel tunables. The options may be indicated to the system through a number of flags. The system administrator 150 also includes a display 210 that allows the system administrator to view a specific tunable, and to view the results of changes to the tunable's parameter values and expressions. The system administrator interface 150 includes a *kctune* command 200 that allows the system administrator to view, and in some cases to change, kernel and user-defined tunables, current values of the tunables, and expressions that are used to compute the values. Besides *kctune,* other system administrator interfaces are available to work with tunables. For example, a web-based graphical user interface, *kcweb*, and a system call, *settune*() for use by programs, are available. The *kctune* command 200 includes a number of options 202, indicated by flags 204, that allow the system administrator to work with the kernel and user-defined tunables. Examples of options are: list tunables and their values (-a): with verbose output (-v), only tunables with changes held for next boot (-d), in a saved configuration (-c); set a tunable value (assign) (-s); hold change until next boot (-h); and create user-defined tunable (-u). The set tunable value (-s) option may be used by the system administrator to define an expression relating a user-defined tunable to one or more kernel tunables. Other options (not shown in Figure 3) may include: list tunables including derived tunables, and grouped by module name; set tunables to default values; increment a

tunable value; apply change to saved kernel configuration; and make sure a tunable value is at least n.

Using the various *kctune* command options 202, the system administrator can see that each tunable has a name and a textual description. Each tunable is associated with a kernel module whose name is listed in a verbose output. Tunables can be seen and changed only if they are associated with a module that is installed on the computer operating system, or are user-defined tunables.

When displaying tunable information for the currently running computer operating system, *kctune* 200 includes the current tunable value and the expression used to compute the value. If changes to the tunable's value are being held for the next system boot, the next boot value and expression are also shown. A verbose listing also shows the tunable's value when the system was last booted.

Tunable values are computed integer expressions, which can refer to other tunable values. The value of a tunable could be 4200, 0x400, or 4*nproc+20. Values and expressions use the syntax of the C programming language. Therefore numbers can be written in decimal, octal, or hexadecimal format.

Kernel tunables can be set to a default value, meaning the system will determine the optimum value for that parameter. In an embodiment, user-defined tunables, if set to default, are removed from the UNIX® operating system.

Tunable values can be assigned based on an expression. For example, user-defined tunable, utunable1, and kernel tunable, ktunable2, may be related by: ktunable2=utunable1*2+100, which means the value assigned to ktunable2 is the value of utunable1 times two, plus 100. Values of other kernel tunable may also be assigned by expressions using utunable1.

User-defined tunables operate in much the same fashion as kernel tunables. To create a user-defined tunable, the system administrator uses *kctune* 200, but with the *–u* (user-defined) flag. Subsequent changes to the user-defined tunable involve use of the *kctune* command 200, but use of the *–u* flag is not required. To remove the user-defined tunable utunable1, the system administrator uses: *kctune –s* utunable1= .

Figure 4 is a logical diagram of a user-defined tunable. Figure 4 shows user-defined tunable 300 including a tunable name 302. The tunable name 302 may incorporate the name of its associated kernel module. The tunable name 302 should be unique as used in the computer system. The user-defined tunable 300 also includes an assigned value 306, which is an integer value. Finally, the user-defined tunable 300

includes an expression 314 that relates the user-defined tunable 300 to other tunables, and in particular to kernel tunables.

Figure 5 is a flowchart illustrating a process 400 for implementing user-defined tunables. The process 400 begins in block 401. In block 410, the system administrator initializes the *kctune* command, and depending on the options selected, will be provided with a display of user-defined and kernel tunables. For example, if the verbose flag (-v) is selected, a detailed description of each tunable is provided. In block 420, the system administrator makes a selection to either create a user-defined tunable or to modify an existing tunable. If the system administrator selects create a user-defined tunable, the process 400 moves to block 430 and the system administrator selects the –u flag to initiate creation of the user-defined tunable. In block 440, the system administrator describes the user-defined tunable, including some or all the fields shown in Figure 3. In block 450, the system administrator selects a save option, such as hold until next boot (-h). The process then moves to block 490 and ends.

In block 420, if the system administrator selects modify an existing tunable, the process 400 move to block 460, and the system administrator designates the tunable to be modified. Using the system administrator interface 150, the system administrator can modify an existing user-defined tunable or an existing kernel tunable. In block 470, the system administrator selects a modification option. For example, the system administrator may select to modify an expression relating a user-defied tunable to one or more kernel tunables. In block 480, the system administrator selects a save option. The process 400 then moves to block 490 and ends.

Figure 6 shows a computer system 500 using a UNIX® operating system, and having the feature of user-defined tunables. To implement user-defined tunables, a computer readable medium 510 is provided with appropriate programming 520, including programming to execute the process 400 shown in Figure 5. The programming 520 works in conjunction with the installed UNIX® operating system to provide the user-defined tunable functionality.

The computer readable medium 500 may be any known medium, including optical discs, magnetic discs, hard discs, and other storage devices known to those of skill in the art. Alternatively, the programming required to implement the user-defined tunables may be provided using a carrier wave over a communications network such as the Internet, for example.